13. 04. 2004

Europäisches Patentamt | European Patent Office | Office européen des brevets

REC'D 21 MAY 2004
WIPO PCT

Bescheinigung Certificate Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr. Patent application No. Demande de brevet n°

03290920.2

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

R C van Dijk

PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

Anmeldung Nr:
Application no.: 03290920.2
Demande no:

Anmeldetag:
Date of filing: 11.04.03
Date de dépôt:

Anmelder/Applicant(s)/Demandeur(s):

Thomson Licensing S.A.
46, quai A.Le Gallo
92100 Boulogne-Billancourt
FRANCE

Bezeichnung der Erfindung/Title of the invention/Titre de l'invention:
(Falls die Bezeichnung der Erfindung nicht angegeben ist, siehe Beschreibung.
If no title is shown please refer to the description.
Si aucun titre n'est indiqué se referer à la description.)

Secure distributed system for management of local network representation within network devices

In Anspruch genommene Priorität(en) / Priority(ies) claimed /Priorité(s) revendiquée(s)
Staat/Tag/Aktenzeichen/State/Date/File no./Pays/Date/Numéro de dépôt:

Internationale Patentklassifikation/International Patent Classification/
Classification internationale des brevets:

H04L29/06

Am Anmeldetag benannte Vertragstaaten/Contracting states designated at date of filing/Etats contractants désignées lors du dépôt:

AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IT LU MC NL PT RO SE SI SK TR LI

FIELD OF THE INVENTION

The invention applies to digital networks, especially when they are dynamical, evolutive, heterogeneous, and when they contain wireless parts.

BACKGROUND ART

Definitions:

A network is dynamic when devices can move, be on / off, be reachable or not.

A network is evolutive when new devices may join the network, older devices may definitively disappear from the network or be stolen.

A network is heterogeneous when not all devices are able to communicate by pairs.

A community is a network composed of devices under the responsibility of a main user. The main user is either a single user or a specific user within a group of persons. Only the main user is able to authenticate against community devices in order to perform the validation operation required by the system.

The frontier of a community is defined following its characteristic properties:

- Any device in the community can verify that it belongs to the community.
- Any device in the community, can verify whether any another device also belongs to the community or does not belong to the community.
- Only the main user can perform frontier operations such as inserting or removing devices from the community.

Prior art

Most prior art comes out from the field of Company Wide Digital Networks, Ad-Hoc Networks (i.e. networks with no pre-existing infrastructure, generally build for the specific use of a group of person – Ad-hoc network duration does not exceed group duration), Digital Home Networks, Wireless and Mobile Networking.

The first communities corresponded to a basic model: the community frontier were identical to network frontier. If a device was reachable through the network, then it was a member of the community. Conversely, any device that was not reachable through the network was not a member of the community.

Such communities exactly correspond to isolated Local Area Networks (LAN) as they were used in companies, before the need to connect un-trusted networks (such as the Internet).

In these communities, the security of the frontier relies on two main factors:

- Only authorized users are able to use a device and the network.
- No un-trusted device can be inserted on the network.

Both factors were enforced by the role of a main user (called a network administrator) and the location of devices and network in a secure place.

These communities are not adapted in cases where the network is mobile or needs to cross un-trusted devices. Administrative tasks are also very demanding, and generally not accessible to a typical domestic main user. Last, the security model is not fault-resistant as all the community is compromised as soon as one of its members is compromised.

When the need for communication over un-trusted networks arose, the former paradigm didn't suffice. Frontier had to be materialized in a different way, that would take in account the possibility to cross non-trusted networks, such as the Internet.

This gave birth to frontier components such as secured routers and firewalls, as well as the notion of private addressing domains. Such components enforce correct frontier properties by allowing or denying cross-frontier access. Typical architecture is a diode firewall allowing outgoing connections and forbidding incoming connections.

The security of the frontier of such community relies mostly on the ability of frontier components to detect whether external connections are authorized or not. Inside the network, the security relies on the same two factors (authorized access and no un-trusted device insertion).

These communities are not adapted in cases where the network is very evolutive or when a lot of devices have a nomadic behavior.

Cross-Network communities really started with nomadic behaviors; when a device needs to access the community from an external network location. Firewall helped enforcing frontier properties, together with authentication servers.

Protocols such as IPv6 (New version of Internet Protocol, as specified in "*RFC 2460 Internet Protocol, Version 6 (IPv6) Specification. S. Deering, R. Hinden. December 1998*") and some VPN (Virtual Private Network) technologies include mobility and security function that help securing

communities frontiers. These include HIP and SUCV (described in "*C. Montenegro and C. Castelluccia. Statistically Unique and Cryptographically Verifiable (SUCV) identifiers and addresses. In NDSS'02, Feb. 2002*"). In this case however, the complexity is not manageable by a typical domestic user. Moreover, these technologies rely on devices homogeneity (for instance: each device has a valid IPv6 address).

F. Stajano proposed a more generic method: the Resurrecting Duckling (in "*F. Stajano The Resurrecting Duckling – What Next? Lecture Notes in Computer Science, 2133:204–211, 2001*" and in "*F. Stajano and R. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In 7th International Workshop on Security Protocols, pages 172–194, 1999.*"). In this method, however, the main user must validate operations whenever a new device is added to the community. Moreover, banishment of a device from the community is not an easy operation in the general case.

The main problems when managing and securing community frontiers are:

- Complexity and lack for user friendliness at least in regard to domestic user needs. This is mostly true for firewalls (even personal firewalls) that remain complex if a fair security level is to be achieved.
- Need for heterogeneity: most existing methods fail when not all devices are able to communicate by pairs.
- Lack of robustness when devices are compromised or stolen. More precisely, a posteriori revocation (banishment) of a device is not a simple action in most existing methods.

SUMMARY OF THE INVENTION

In order to overcome the above-mentioned problems, the invention proposes a system for the secure and distributed management of a local network representation within network devices, characterized in that each network device (x) contains :

a provable identity (id_x) or means to generate or to obtain a provable identity;

objects (MT(x), UT(x), DT(x)) memorizing trust relationships between devices of the community; and

means for establishing a protocol for trust relationships synchronization.

BRIEF DESCRIPTION OF THE DRAWINGSThe various features and advantages of the present invention and its preferred embodiments will now be described with reference to the accompanying drawings which are intended to illustrate and not to limit the scope of the present invention and in which:

Fig. 1 illustrates parts of a device implementing the invention.

Fig. 2 illustrates an example of a community created according to the invention.

Fig. 3 to 7 illustrate a flowchart of the preferred protocol executed in a device z according to the invention.

Fig. 8 to 12 are temporal diagrams illustrating different possible situations between devices implementing the protocol illustrated in Fig. 3 to 7.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTSIn the following of the description, the following notation will be used:

x, y, z, t, j	Variable names for devices.
id_x	Provable identity of device x.
Λ	Community of devices.
MT(x)	Set of devices trusted by x AND trusting x.
UT(x)	Set of devices trusted by x.
DT(x)	Set of devices distrusted by x.
$S_x(id_y)$	Proof that device y is trusted by device x.

The invention is based on the following elements:

1. Each device x of the community has a provable identity id_x or is able to generate or to receive a provable identity.
2. Each device x of the community memorizes trust relationships between devices of the community in objects MT(x), UT(x) and DT(x) respectively containing:
   - MT(x): a set of devices trusted by x AND trusting x.
   - UT(x): a set of devices trusted by x.
   - DT(x): a set of devices distrusted by x.
3. Each device x of the community furthermore memorizes proofs $S_j(id_x)$ received from other devices j of the community that x is trusted by j.

4. A protocol for trust relationships synchronization is implemented in each device of the community.
5. The user has the possibility to validate or invalidate trust relationships between some devices.

First, the invention allows distributed and secure enforcement of community frontiers.

Second, the invention minimizes number and complexity of interaction between community devices and the main user.

In the following of the description, the following notation will be used:

x, y, z, t	Variable names for devices.
id_x	Provable identity of device x.
Λ	Community of devices.
MT(x)	Set of devices trusted by x AND trusting x.
UT(x)	Set of devices trusted by x.
DT(x)	Set of devices distrusted by x.
$S_x(id_y)$	Proof that device y is trusted by device x.

Preferably, objects MT(x), UT(x) and DT(x) are implemented by lists containing provable identities id_j of the devices j which are part of the set.

For example if a device x trusts a device y and is trusted by y, MT(x) will contain id_y. MT(x) may also possibly contain some cryptographic material, such as keys to allow devices of the community to securely exchange data. In the above example, MT(x) may contain a symmetrical key K_{xy} shared between devices x and y.

In a variant embodiment of the invention, the list of proofs $S_j(id_x)$ may be stored in MT(x), each proof $S_j(id_x)$ being stored with the identity id_j of the device trusting x and trusted by x.

In the same way, if a device x trusts a device z but is not necessarily trusted by z, then UT(x) will contain id_z. UT(x) may also contain some cryptographic material.

DT(x) also contains identities id_j of devices j which are distrusted by x. It may also possibly contain other data such as cryptographic material.

Basic community operations are:

- Initialization of a community, denoted **init**:
  The **init** operation corresponds to the creation of the community, generally with a single device.

- Insertion of device in a community, denoted **insert**:
  The **insert** operation occurs when a new device enters the community. This new device should be able to identify the other devices of the community as belonging to the community and the other members of the community should identify the new device as a member of the community.
- Removal of a device from a community, denoted **remove**.
  The **remove** operation shall be used when a device is obsolete. This operation will extract the device from the community, but will not modify trusts relations. In particular, in the case when two devices y and z build a trust relationship upon the assumption that both devices have trust relationships with device x, the fact that device x has been removed has no impact.
  Then the **remove** operation does not require any information transmission with other community devices. In particular, this operation is valid in the case of a single device community.
  Removing a device x consists in:
  - Destroying x identity (id_x) and the ability for x to prove this identity.
  - Resetting all trust relationships, that is making all sets MT(x), UT(x), DT(x) empty.

  After removal, the device x is unable to broadcast its identity (which has been destroyed). He cannot take any part in community devices transmissions as no community device accepts a transmission with unidentified device.
- Banishment of a device from a community, denoted **banish**.
  The **banish** operation shall be used when a device has been lost or stolen, or when a device is resale to another user, from another community. In this case, the device itself is not available. Moreover, new trusts relationships that can be build upon trusts assumptions with the banished device shall become impossible.
  To banish a device x, the user must select another available device y that already has a trust relationship with x (i.e. its identity id_x belongs either to UT(y) or MT(y)). The user asks y to add $\{id_x\}$ in its list of distrusted devices DT(y).
  The synchronization operation will insure diffusion of the information that device x is distrusted. Depending on how often

devices of the community interact, this information may diffuse faster over some devices, and slower over all devices.

Figure 1 illustrates which elements are contained in a device for implementing the invention.

A device x typically contains a CPU (Central Processing Unit), a User Interface, a memory for storing objects MT(x) UT(x) and DT(x) as well as the list of proofs $S_j(id_x)$ received from other devices j of the community that x is trusted by j. The device furthermore contains at least one network interface for communication with other devices of the community. One device may contain several network interfaces in order to allow heterogeneous communications in the community.

Figure 2 illustrates an example of a community of devices represented by a multi-site domestic network. Devices are for example a Personal Computer, a TV set, a storage unit, a PDA, etc. In the situation of figure 2, we suppose that all trusts relationships between devices are mutual trust. Figure 2 illustrates the moment when device c is about to accept new device d in the community, with user validation.

In the preferred embodiment of the invention, each device contains a local agent responsible for its security. The first task of the agent is to manage its own provable identity. A provable identity is an identity that has the property of being able to be checked by anyone, while being very hard to impersonate. For instance, the public key of a public/private key pair is a provable identity: an agent pretending being identified by its public key can prove it by signing a challenge with its private key. SUCV is another mechanism designed for IP networks based on the idea of provable identity.

The local agent is in charge of generating, escrowing and endorsing its provable identity that will be used to authenticate itself in front of the other devices of the community.

The agent is also in charge of locally authenticating the user who makes authority on the device to ensure that the security-relevant requests are legitimate. This local authentication is totally independent from its own provable identity as well as from the keying process that is made between devices. As a consequence, each device can have its own best suited authentication procedure (for example by entering a PIN on the device or by biometrics).

Finally, the agent is in charge of community management. It possesses and maintains its own list of the community members which are stored in objects MT, UT and DT describes above. Depending on the implementation chosen, these objects can be stored in a single list or in different lists. This list or theses lists describe(s) the local knowledge the agent has of its community. By securely updating the content of objects MT, UT and DT, an agent manages its community.

Objects MT, UT and DT can be updated by two different means: an agent trusts its owner (i.e. the user who owns the device) to decide which device can enter in its community. It also trusts the agents it knows as belonging to its community (i.e. the agents having their provable identity in its MT or UT), to introduce to him new members of the community. Agents belonging to the same community "synchronize" their information with each other in a secure way to maintain their respective objects MT, UT and DT up to date.

The agent can be physically implemented in several different way.

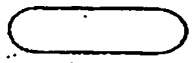
It may be a software downloaded or embedded in the device. It can also be a software running in a smart card inserted in the device. The agent can also be implemented by a chip or chip set containing a software.

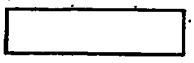
We will now describe more precisely the protocol which is implemented in a device z according to the invention. This protocol is described in view of figures 3 to 7.

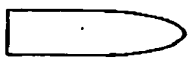
In addition to the notation previously described, the following notations are used in these figures:

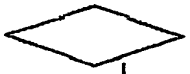
∧ Logical AND

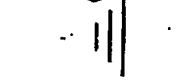
∃?y, P(y) Does it exist a device y, such that condition P holds for y

 Start point

 Sequence instructions

 Timeouted intruction (returns to step 3 unless otherwise specified)

 Binary condition

 End point

1: Start point used when the main user just acquired a device z with no *id*.

2: All necessary operations for device z initialization, including: software code insertion (unnecessary for chip implementation), creation of cryptographic key material, creation of the provable identity id_z of the device z, set up of lists MT(z), UT(z) and DT(z) to *empty*.

3: Normal start point for an already initialized device z.

100: Contains all operations and conditions necessary for device z to detect whether another device t belong to the same community Λ or not. Detail for these operations are given in points 101 to 104.

101: Device z sends information by any available mean (including wired or wireless network protocols). The broadcast information is: id_z and MT(z). The next activated point will be 102.

102: This point is automatically activated after the point 101. Device z waits and listens to all its network interfaces, until it obtains id_t and MT(t) from a device t (case 1) or until a timeout expires (case 2). Typical timeout duration in the case of domestic network is one or two minutes. If the case 1 occurs then the next point activated will be 103 else (case 2) it is 101.

103: This point is activated if the information id_t and MT(t) have been received from a device t. Device z verifies if it distrusts t or not. If so, the next activated point will be 3, else it will be 104.

104: This point is activated if device z does not distrust device t. Device z verifies if id_t belongs to MT(z) and if device id_z belongs to MT(t). If both verifications are successful then the next activated point will be 400; else it will be 200.

200: This point is activated if device z has detected that a device t does not (already) belong to the same community. This point contains all operations and conditions necessary for device z to detect whether it can enter the same community as the device t 's one. Detail for these operations are given in points 201 to 209.

201: Device z verifies if it exists a device x such that id_x belongs to the intersection of the lists MT(z) and MT(t). If so the next activated point will be 202 else it will be 204.

202: This point is activated if it exists a device x such that id_x belongs to the intersection of the lists MT(z) and MT(t). Device z asks device t for $S_x(id_t)$. The next activated point will be 203. Point 202 is under the control of a timeout of typical duration 1 minute. If timeout expires, the next activate point is 3.

203: This point is automatically activated after the point 202. Device z receives $S_x(id_t)$. from t and verifies it. If the verification is successful then the next activated point will be 300 else it will be 3.

204: This point is activated if it does not exist any device x such that id_x belongs to the intersection of the lists MT(z) and MT(t). Device z verifies if it exists a device x such that id_x belongs to the intersection of the lists UT(z) and MT(t). If so the next activated point will be 205 else it will be 209.

205: This point is activated if it exists a device x such that id_x belongs to the intersection of the lists UT(z) and MT(t). Device z asks device t for $S_x(id_t)$. The next activated point will be 206. Point 205 is under the control of a timeout of typical duration 1 minute. If timeout expires, the next activate point is 3.

206: This point is automatically activated after the point 205. Device z receives $S_x(id_t)$. from t and verifies it. It the verification is successful then the next activated point will be 207 else it will be 3.

207: This point is activated if device z has successfully verified $S_x(id_t)$. Device z asks device t for UT(t). The next activated point will be 208. Point 207 is under the control of a timeout of typical duration 1 minute. If timeout expires, the next activate point is 3.

208: This point is automatically activated after the point 207. Device z verifies if it exists a device y such that id_y belongs to the intersection of the lists UT(t) and MT(z). If so the next activated point will be 300 else it will be 3.

209: This point is activated if it does not exist any device x such that id_x belongs to the intersection of the lists UT(z) and MT(t). A user validation is requested to activate to the next point 300. Point 202 is under the control of a timeout of typical duration 1 minute, but the main user can configure this duration. If timeout expires, the next activate point is 3.

300: This point is activated when device z has a proof that it can accept the device t in its community. This point contains all operations and conditions necessary for device z to accept device t in its community. Detail for these operations are given in points 301 to 303.

301: Lists UT(z) and MT(z) are updated as follows: id_t is removed from to UT(z) and is inserted in MT(z). The next activated point will be 302.

302: This point is automatically activated after the point 301. Device z sends $S_z(id_t)$ to t. The next activated point will be 303.

303: This point is automatically activated after the point 303. Device z waits for $S_z(id_t)$. from t. The next activated point will be 400. Point 303 is under the control of a timeout of typical duration 1 minute. If timeout expires, the next activate point is 3.

400: This point is automatically activated after the points 104 (devices *z* and *t* already belong to the same community) or 303, (device *z* has a proof that it can accept the device *t* in its community). This point contains all operations and conditions necessary for device *z* and device *t* to share and update community information. Detail for these operations are given in points 401 to 402.

401: Lists DT(*z*) and UT(*z*) are updated as follows: elements of DT(*t*) are added to DT(*z*), elements of MT(*t*) are added to UT(z), elements of DT(*t*) are removed from to UT(*z*). The next activated point will be 402.

402: This point is automatically activated after the point 401. Device *z* provides device *t* with all the community information it possesses. The next activated point will be 3.

Figure 8 illustrates the operations when a device *b* enters device *a*'s community.

Figure 9 illustrates the operations when device *a* meets device *d* and when *d* enters in *a*'s community.

Figure 10 illustrates the operations when a device c enters in the community of device b.

Figure 11 illustrates the operations when devices c and d establish a trusted relationship without any user interaction (using steps 204 to 208 in Fig. 5).

Figure 12 illustrates the operations when devices a and c establish a trusted relationship without any user interaction (using steps 201 to 203 in Fig. 5).

The invention presents the following advantages.

This invention applies to communities that are highly dynamic, evolutive and heterogeneous. Prior art solutions do not apply in such cases or are very demanding to the main user, who is rather a network administrator than for instance a domestic user.

The invention allows secure distribution of any information relevant to the community. These include, but are not limited to: configuration information, time and time-stamping information, third party protocol keys, third party mobile agents, antiviral signature files...

The invention applies various technologies, as the agent can be inserted in most type of networking devices.

The invention applies to previously constituted communities, as well as it applies to newly constituted communities: the agent can be inserted in

older devices if the support enough computation and memory capacities. This includes, but do not restrict to, older personal computers and older Personal Digital Agents.

The invention allows simple banishment of a lost, stolen or compromised device. Other state of the art solutions don't provide easy means for banishing a device that is not accessible anymore.

The invention insures correct information synchronization and diffusion between community devices. This point allows transmission of third party cryptographic material for use by other protocols or system. As a non-limitative list of examples, the invention can transmit:

- Shared secrets for use as WEP or IPSec keys
- Cryptographic digest of files that will be transmitted over possibly insecure protocols (such as FTP). This files may be software patches, virus lists, automated security procedures...
- Cryptographic signature of new versions of software agents (as the one used by the invention).

CLAIMS

1. System for the secure and distributed management of a local network representation within network devices, characterized in that each network device (x) contains :

a provable identity (id_x) or means to generate or to obtain a provable identity;

objects (MT(x), UT(x), DT(x)) memorizing trust relationships between devices of the community; and

means for establishing a protocol for trust relationships synchronization.

ABSTRACT

Secure distributed system for management of local network representation within network devices

Figure .

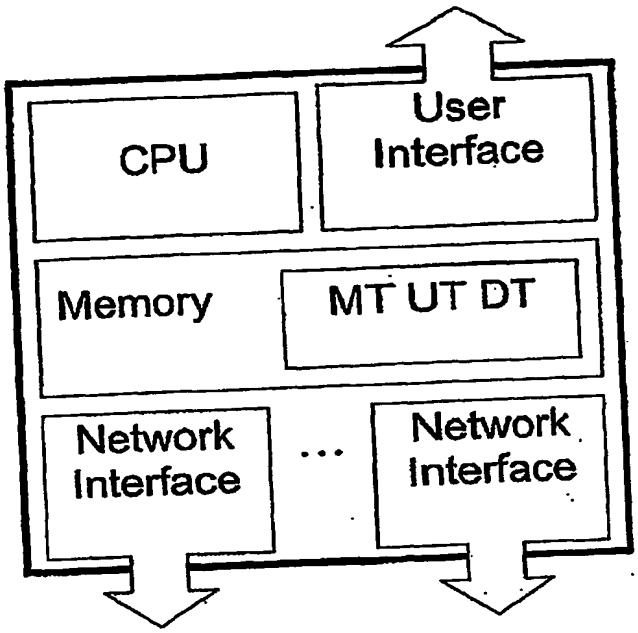

CPU
User Interface
Memory
MT UT DT
Network Interface
...
Network Interface

Fig. 1

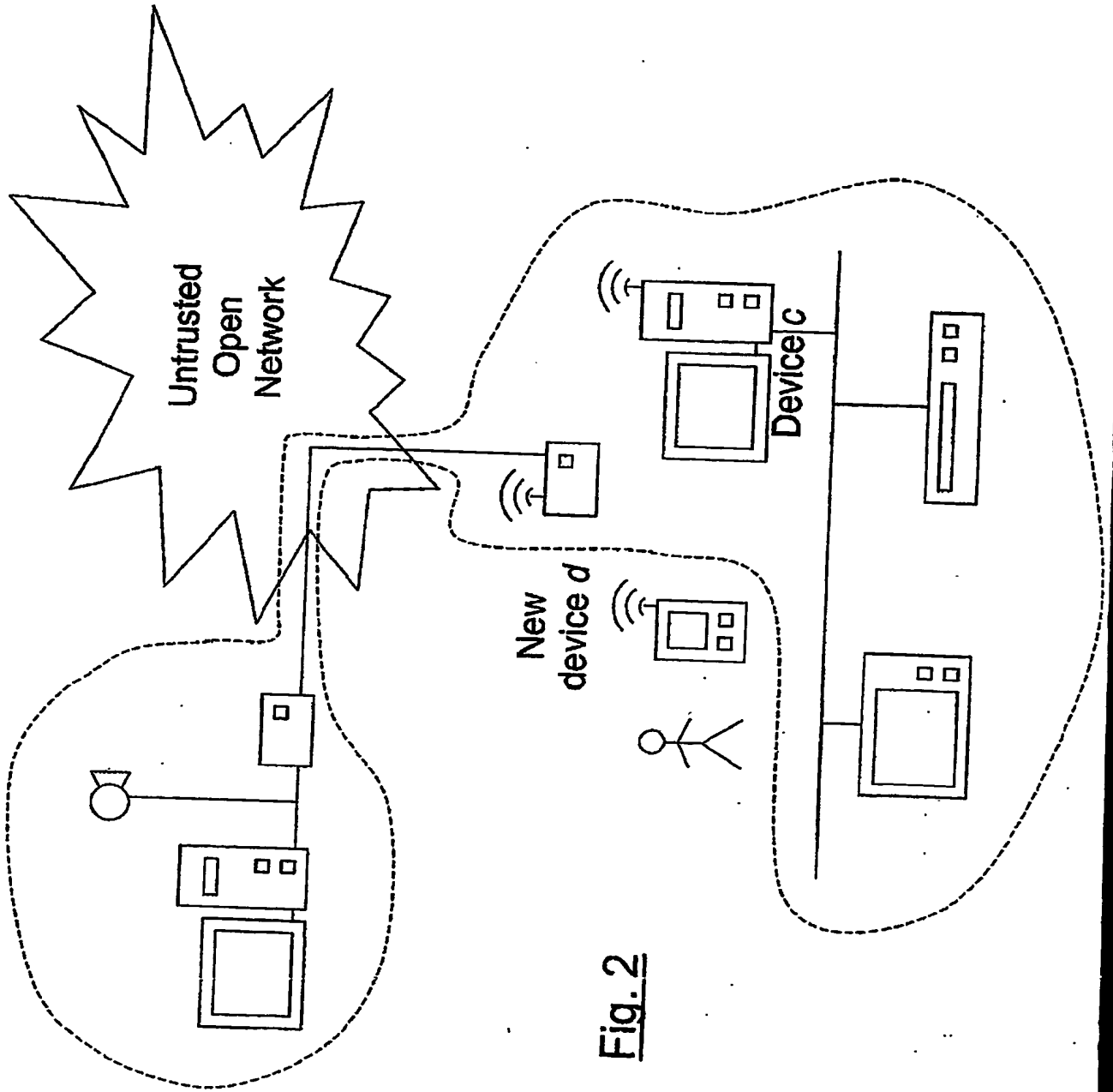


Fig. 2

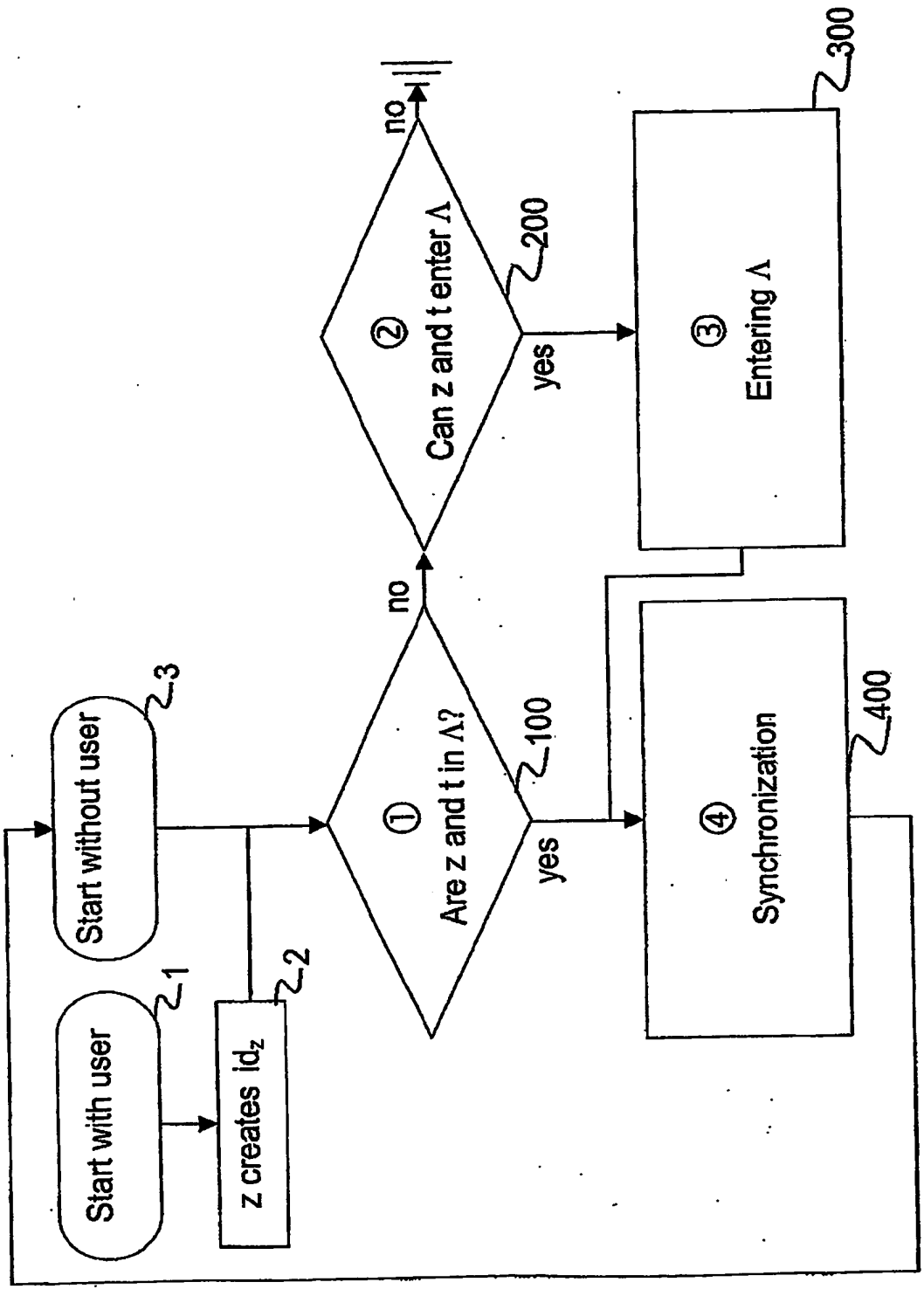

Start with user
1
z creates id_z
2
Start without user
3
① Are z and t in Λ?
100
no
yes
② Can z and t enter Λ
200
no
yes
③ Entering Λ
300
④ Synchronization
400

Fig. 3

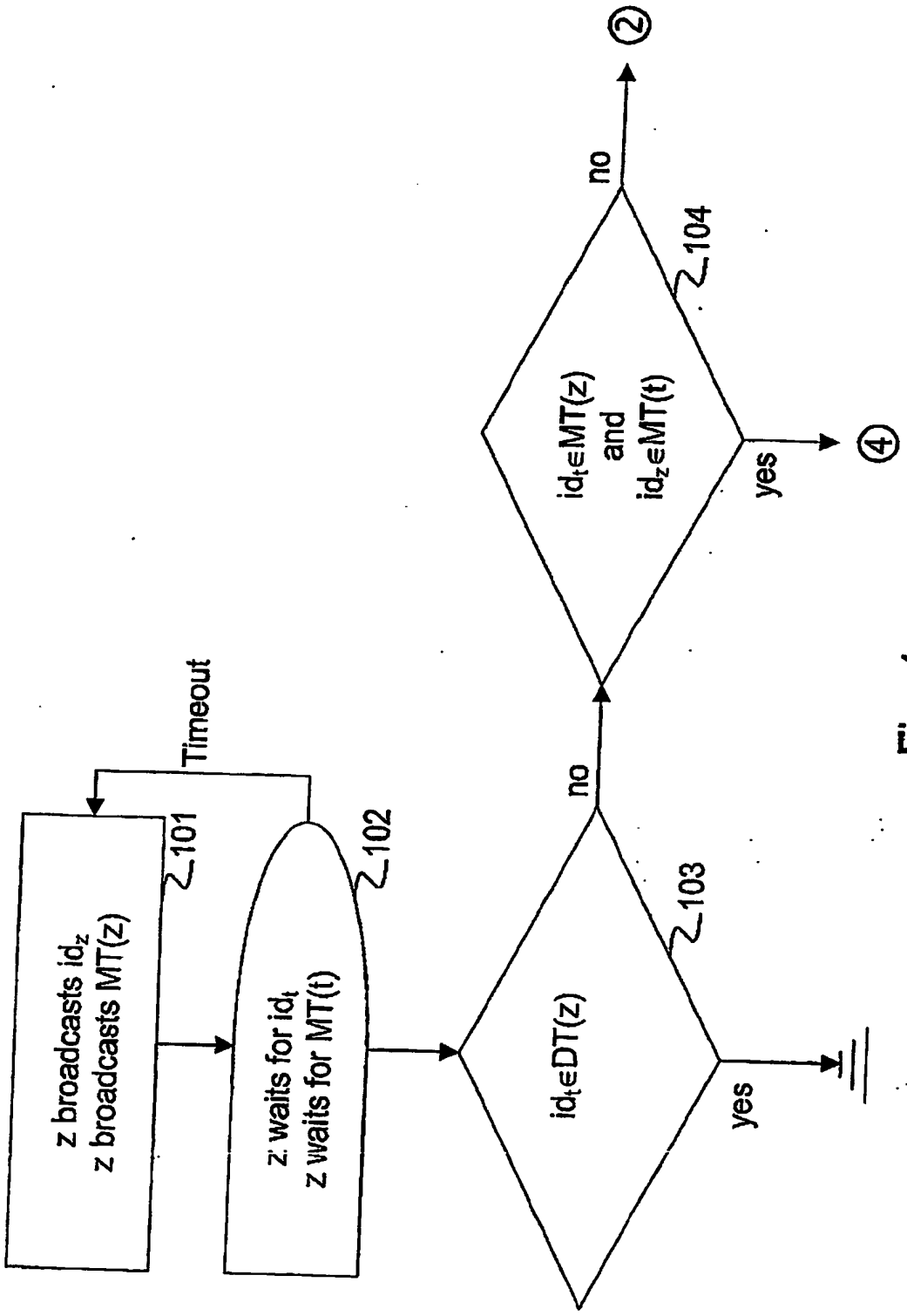

z broadcasts id_z
z broadcasts MT(z)
101
Timeout
z' waits for id_t
z waits for MT(t)
102
$id_t \in DT(z)$
103
no
yes
$id_t \in MT(z)$ and $id_z \in MT(t)$
104
no
yes
②
④

Fig. 4

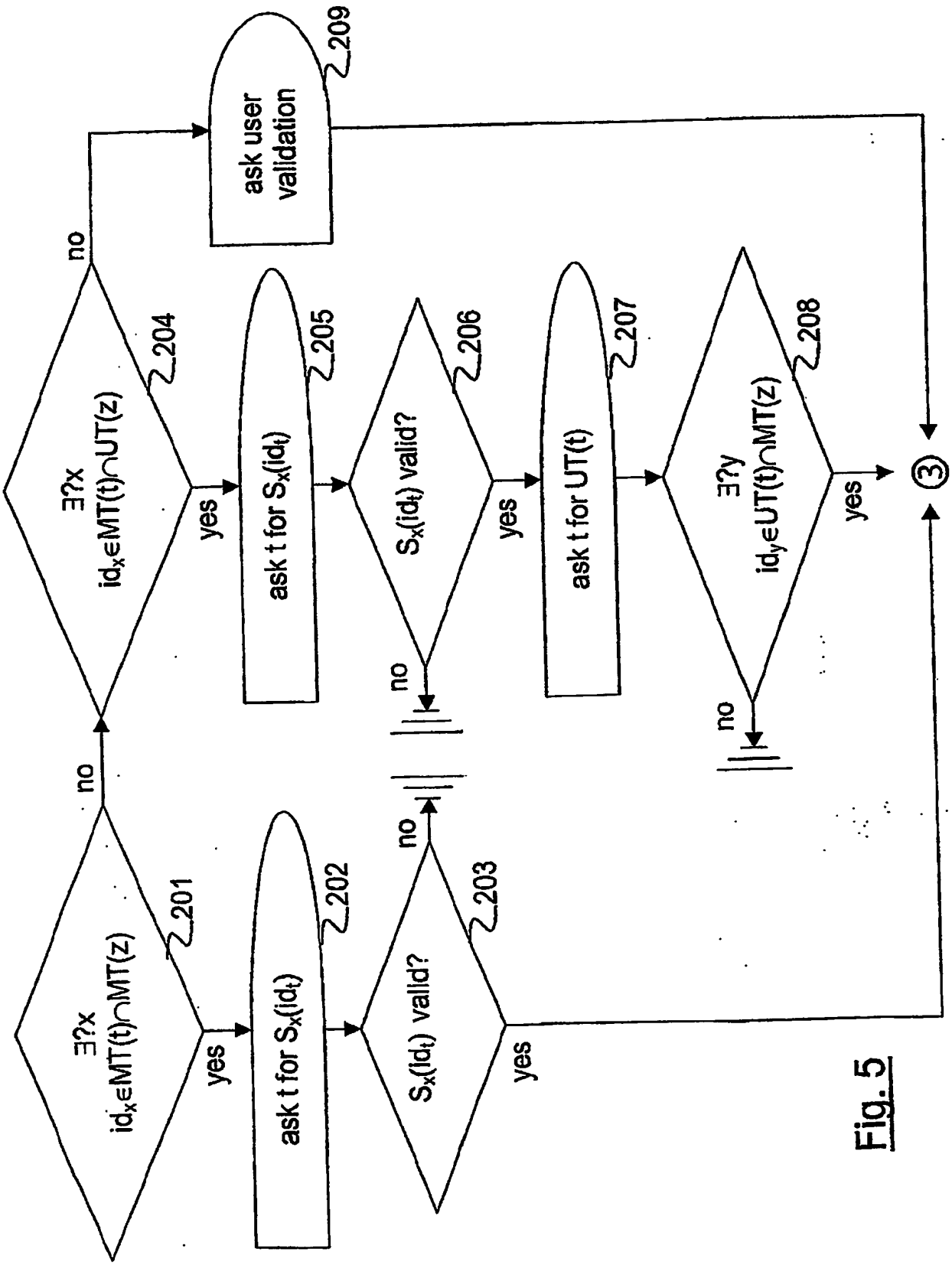

∃?x
$id_x \in MT(t) \cap MT(z)$
201
yes
ask t for $S_x(id_t)$
202
$S_x(id_t)$ valid?
203
no
yes
no
∃?x
$id_x \in MT(t) \cap UT(z)$
204
yes
ask t for $S_x(id_t)$
205
$S_x(id_t)$ valid?
206
no
yes
ask t for UT(t)
207
∃?y
$id_y \in UT(t) \cap MT(z)$
208
no
yes
no
ask user validation
209
③

Fig. 5

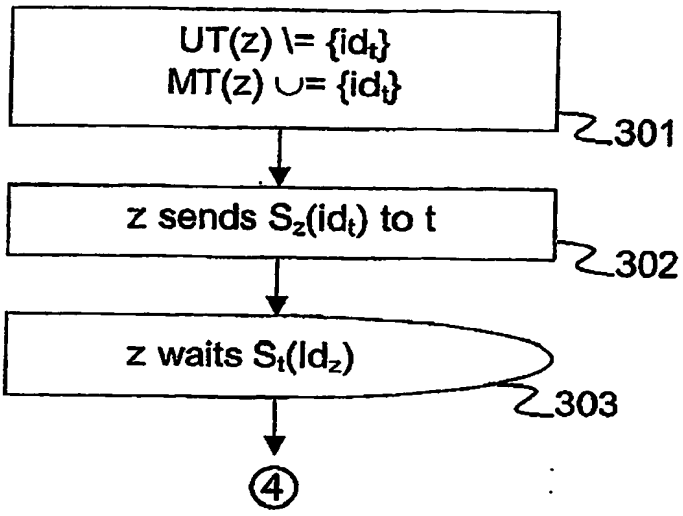

UT(z) \= {id_t}
MT(z) ∪= {id_t}
301
z sends S_z(id_t) to t
302
z waits S_t(Id_z)
303
④

Fig. 6

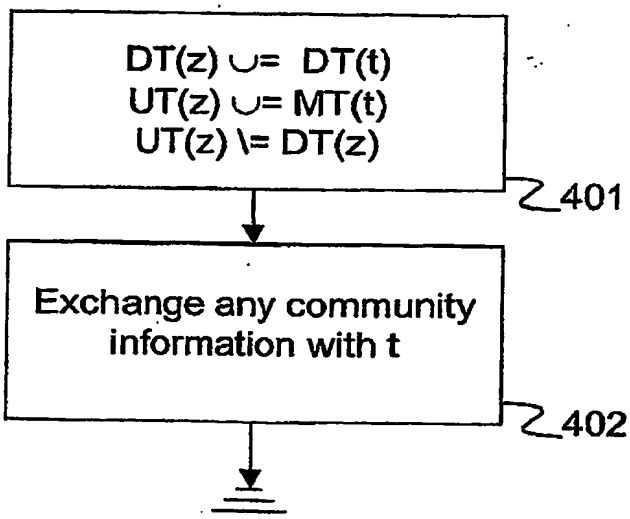

DT(z) ∪= DT(t)
UT(z) ∪= MT(t)
UT(z) \= DT(z)
401
Exchange any community information with t
402

Fig. 7

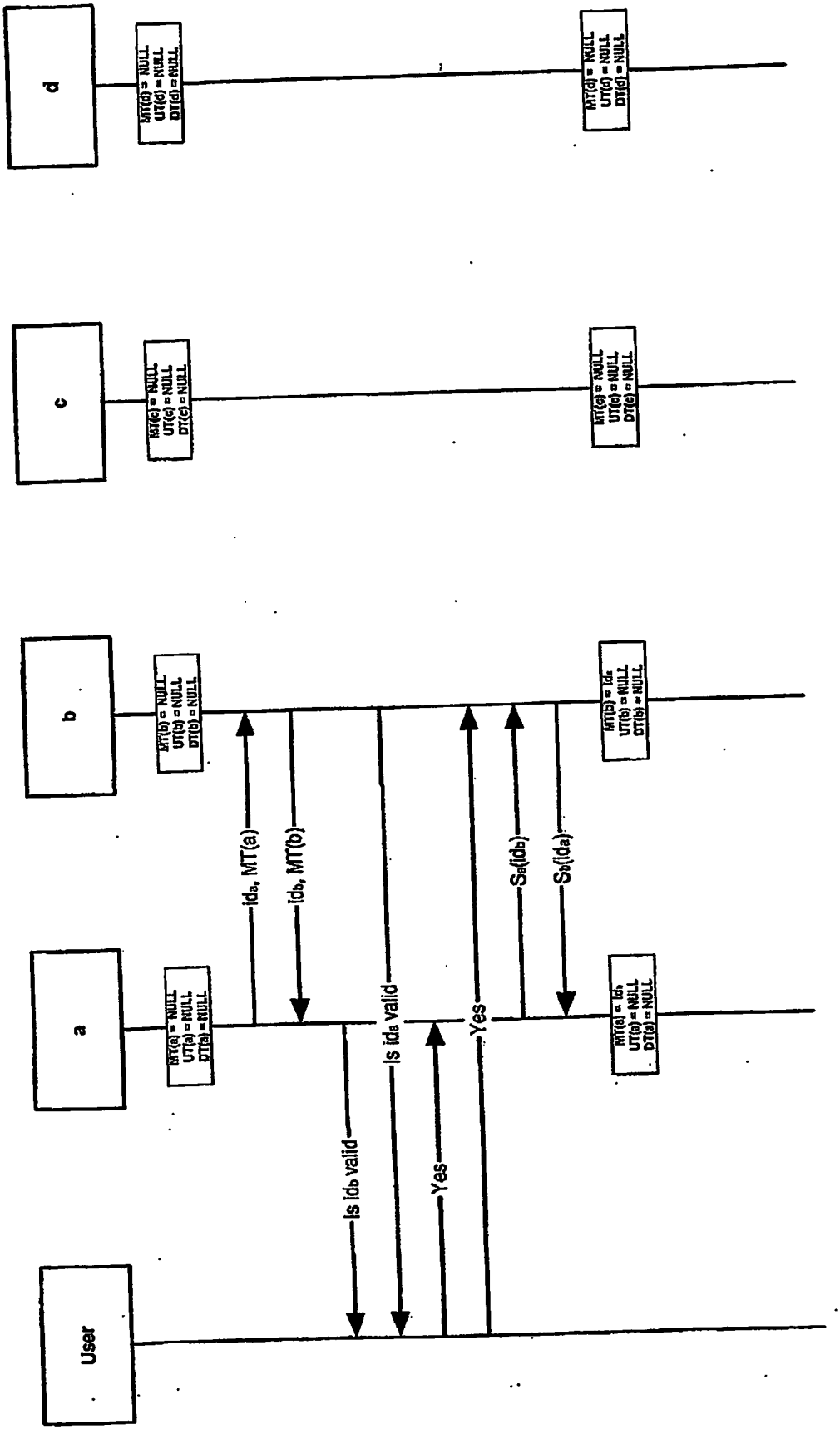

User
a
b
c
d
MT(a) = NULL
UT(a) = NULL
DT(a) = NULL
MT(b) = NULL
UT(b) = NULL
DT(b) = NULL
MT(c) = NULL
UT(c) = NULL
DT(c) = NULL
MT(d) = NULL
UT(d) = NULL
DT(d) = NULL
MT(b) = Ida
UT(b) = NULL
DT(b) = NULL
MT(a) = Idb
UT(a) = NULL
DT(a) = NULL
Ida, MT(a)
Idb, MT(b)
Is Idb valid
Is Ida valid
Yes
Yes
Sa(Idb)
Sb(Ida)

Fig. 8

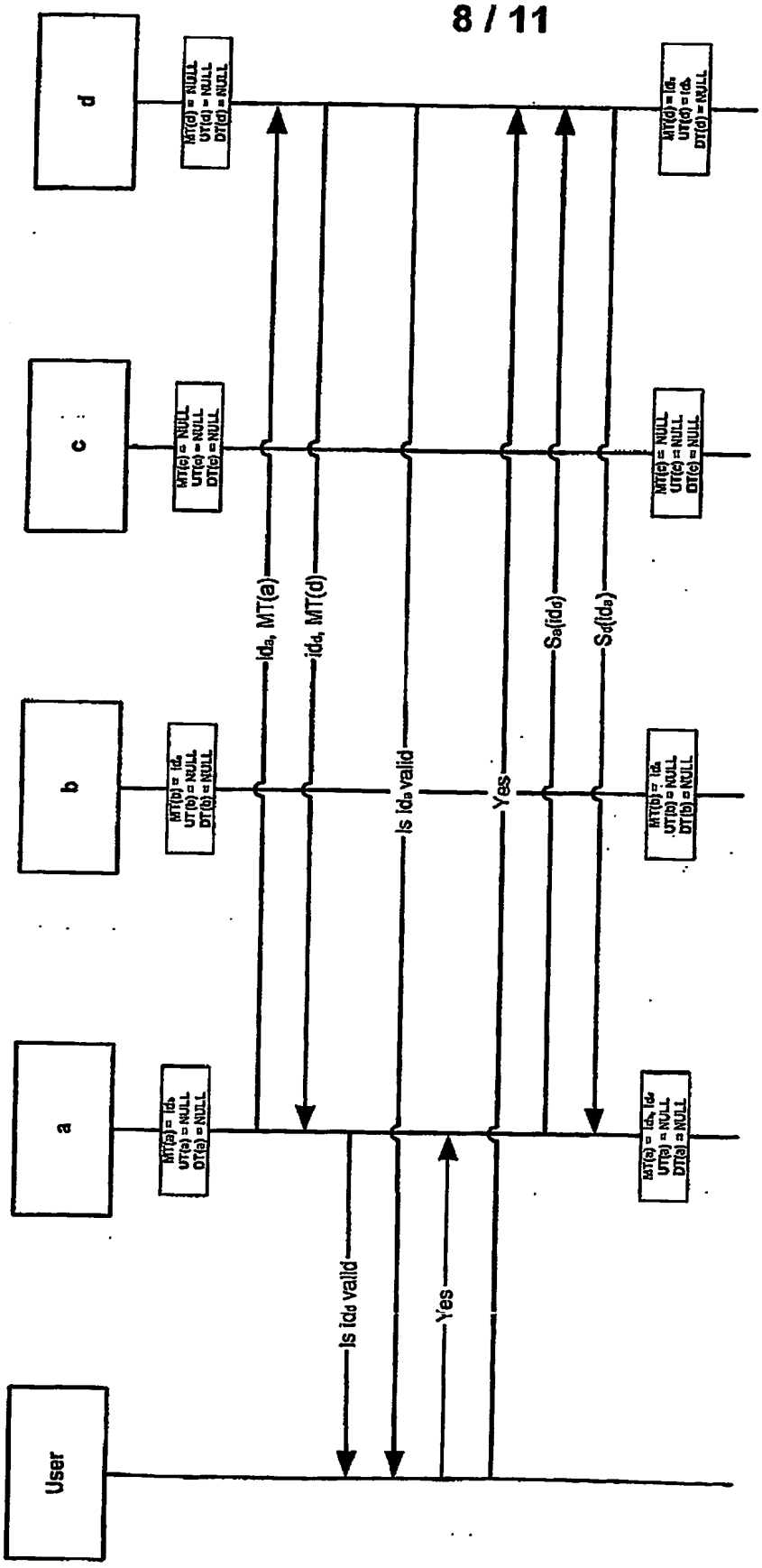

d
c
b
a
User
MT(d) = NULL
UT(d) = NULL
DT(d) = NULL
MT(d) = idd
UT(d) = idd
DT(d) = NULL
MT(c) = NULL
UT(c) = NULL
DT(c) = NULL
MT(c) = NULL
UT(c) = NULL
DT(c) = NULL
MT(b) = idb
UT(b) = NULL
DT(b) = NULL
MT(b) = idb
UT(b) = NULL
DT(b) = NULL
MT(a) = ida
UT(a) = NULL
DT(a) = NULL
MT(a) = ida, idd
UT(a) = NULL
DT(a) = NULL
ida, MT(a)
idd, MT(d)
Is ida valid
Yes
Sa(idd)
Sd(ida)
Is idd valid
Yes

Fig. 9

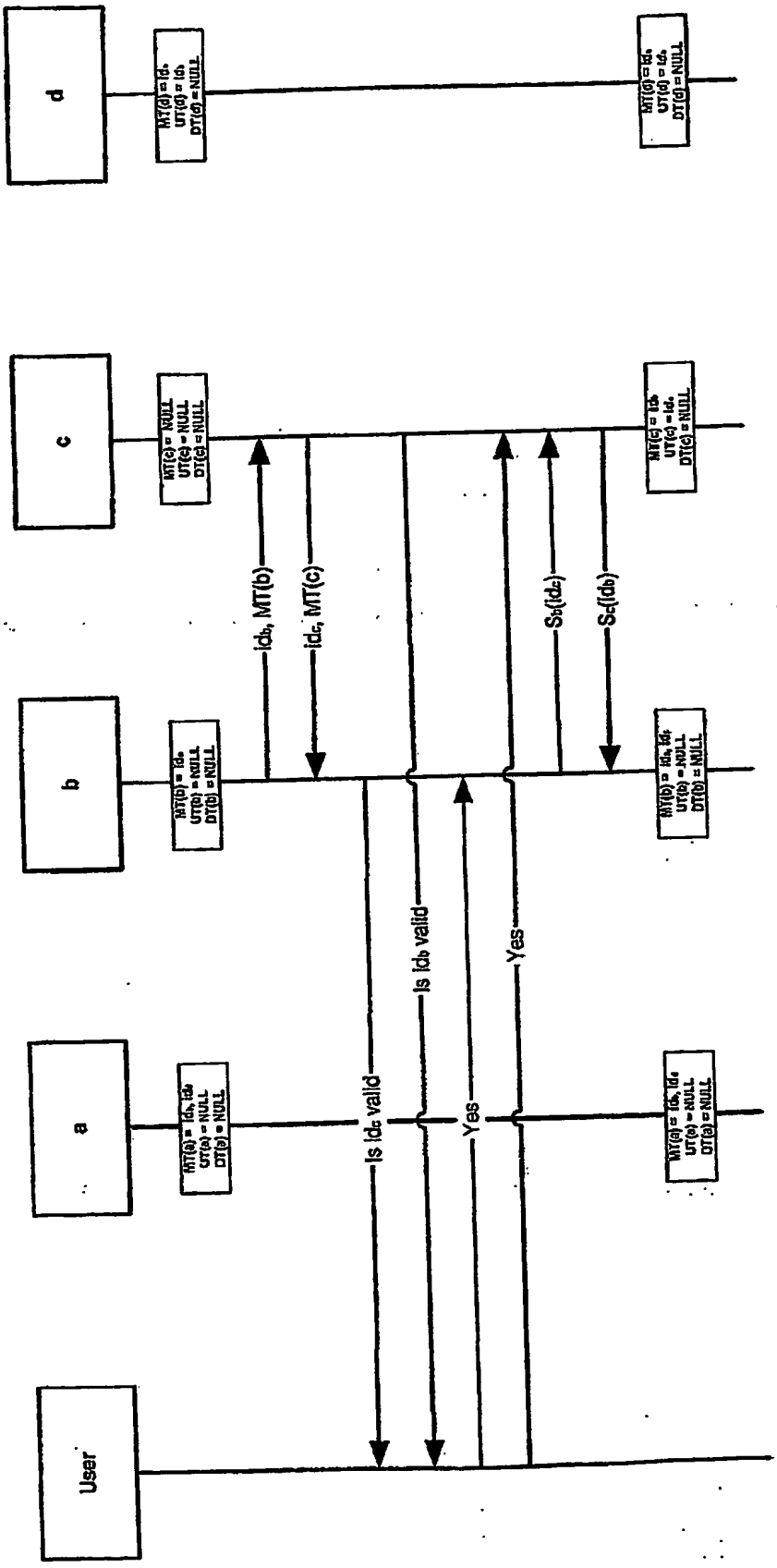

d
c
b
a
User
Is Idc valid
Is Idb valid
Yes
Yes
Idb, MT(b)
Idc, MT(c)
Sb(Idc)
Sc(Idb)

Fig. 10

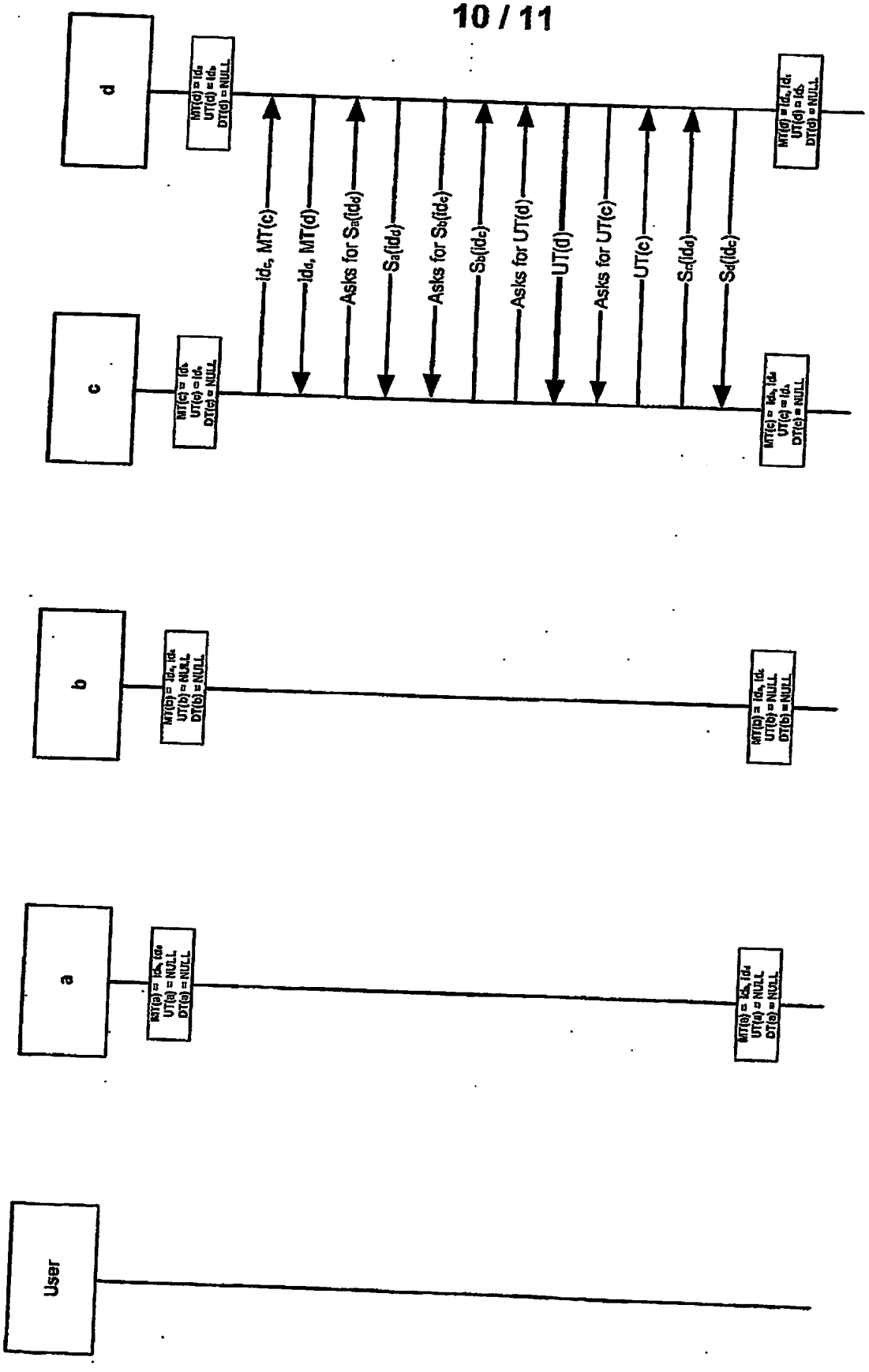

User
a
b
c
d
MT(a) = Id_a, Id_b
UT(a) = NULL
DT(a) = NULL
MT(b) = Id_a, Id_b
UT(b) = NULL
DT(b) = NULL
MT(c) = Id_c
UT(c) = Id_c
DT(c) = NULL
MT(d) = Id_d
UT(d) = Id_d
DT(d) = NULL
Id_c, MT(c)
Id_d, MT(d)
Asks for S_a(Id_d)
S_a(Id_d)
Asks for S_b(Id_c)
S_b(Id_c)
Asks for UT(d)
UT(d)
Asks for UT(c)
UT(c)
S_c(Id_d)
S_d(Id_c)
MT(a) = Id_a, Id_d
UT(a) = NULL
DT(a) = NULL
MT(b) = Id_a, Id_d
UT(b) = NULL
DT(b) = NULL
MT(c) = Id_d
UT(c) = Id_d
DT(c) = NULL
MT(d) = Id_d, Id_c
UT(d) = Id_c
DT(d) = NULL

Fig. 11

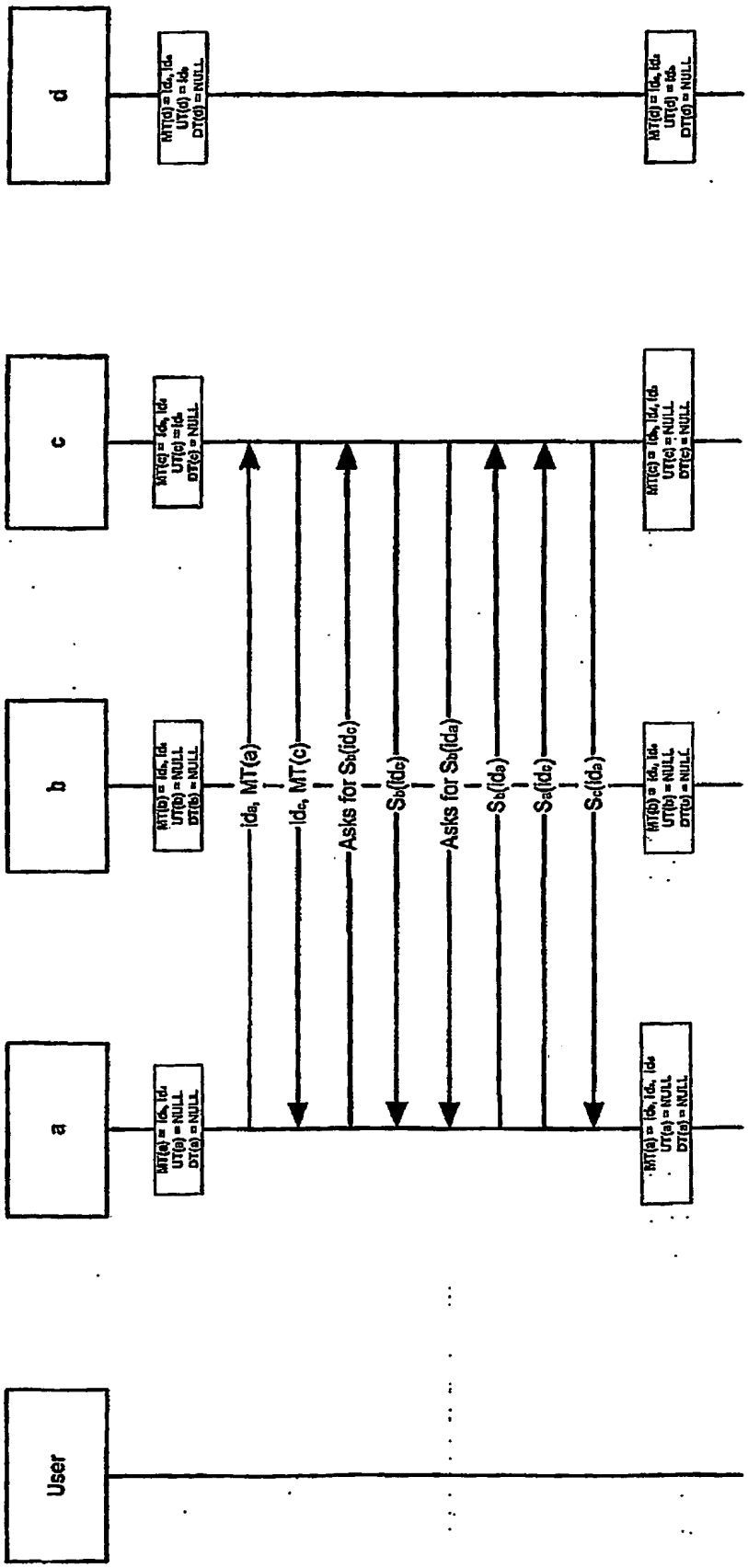

User
a
b
c
d
MT(a) = Id_a, Id_d
UT(a) = NULL
DT(a) = NULL
MT(b) = Id_b, Id_d
UT(b) = NULL
DT(b) = NULL
MT(c) = Id_c, Id_d
UT(c) = NULL
DT(c) = NULL
MT(d) = Id_d, Id_d
UT(d) = Id_d
DT(d) = NULL
Id_a, MT(a)
Id_c, MT(c)
Asks for S_b(Id_c)
S_b(Id_c)
Asks for S_b(Id_a)
S_b(Id_a)
S_a(Id_c)
S_c(Id_a)

Fig. 12

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- [x] BLACK BORDERS
- [x] IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- [x] FADED TEXT OR DRAWING
- [ ] BLURED OR ILLEGIBLE TEXT OR DRAWING
- [ ] SKEWED/SLANTED IMAGES
- [x] COLORED OR BLACK AND WHITE PHOTOGRAPHS
- [ ] GRAY SCALE DOCUMENTS
- [ ] LINES OR MARKS ON ORIGINAL DOCUMENT
- [ ] REPERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- [ ] OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning documents *will not* correct images problems checked, please do not report the problems to the IFW Image Problem Mailbox